AFRL-HE-WP-TP-2006-0008

AIR FORCE RESEARCH LABORATORY

Image Based Tracking System

Vincent M. Parisi

Human Effectiveness Directorate
Warfighter Interface Division
Wright-Patterson AFB OH 45433

January 2006

Human Effectiveness Directorate
Warfighter Interface Division
Wright-Patterson AFB OH 45433

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
January 2006	Technical Paper	

4. TITLE AND SUBTITLE
Image Based Tracking System

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Vincent M. Parisi

5d. PROJECT NUMBER
7184

5e. TASK NUMBER
11

5f. WORK UNIT NUMBER
21

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) AND ADDRESS(ES)

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Materiel Command
Air Force Research Laboratory
Human Effectiveness Directorate
Warfighter Interface Division
Wright-Patterson AFB OH 45433-7022

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/HECV

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-HE-WP-TP-2006-0008

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
The clearance number is AFRL/WS-06-0151 and was cleared 18 January 2006.

14. ABSTRACT
This thesis represents the capstone of five years combined academic work by Mr. Kornbau at Kettering University and job experience at AFRL/HECV.

15. SUBJECT TERMS
Heads-up-displays (HUD)

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
Unclassified					Vincent M. Parisi
a. REPORT	b. ABSTRACT	c. THIS PAGE	SAR	57	19b. TELEPHONE NUMBER *(include area code)*
UNC	UNC	UNC			(937) 255-8885

IMAGE BASED TRACKING SYSTEM

A thesis written at

AIR FORCE RESEARCH LABORATORIES

and submitted to

KETTERING UNIVERSITY

in partial fulfillment
of the requirements for the
degree of

BACHELOR OF SCIENCE IN ELECTRICAL ENGINEERING

by

NATHAN T. KORNBAU

December 2005

_____
Author

_____
Employer Advisor

_____
Faculty Advisor

DISCLAIMER

This thesis is submitted as partial and final fulfillment of the cooperative work experience requirements of Kettering University needed to obtain a Bachelor of Science in Electrical Engineering Degree.

The conclusions and opinions expressed in this thesis are those of the writer and do not necessarily represent the position of Kettering University or The United States Air Force Research Laboratories, or any of its directors, officers, agents, or employees with respect to the matters discussed.

PREFACE

This thesis represents the capstone of my five years combined academic work at Kettering University and job experience at The United States Air Force Research Laboratories (AFRL). Academic experience in electrical engineering and engineering design, proved to be valuable assets while I developed this thesis and addressed the problem it concerns.

Although this thesis represents the compilation of my own efforts, I would like to acknowledge and extend my sincere gratitude to the following persons for their valuable time and assistance, without whom the completion of this thesis would not have been possible:

1. Dr. Douglas Melton, Associate Professor of Electrical Engineering at Kettering University, for his support and guidance in the organization and development of this thesis, while acting as my faculty advisor.

2. Mr. Vincent Parisi, Supervisor at AFRL Human Effectiveness Directorate, for all his help and guidance with regard to the technical aspects of this thesis, while acting as my employer advisor.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Equations **Page**

I. INTRODUCTION

This chapter will provide a general overview of project including problem description, necessary back ground information, and the methodology used to address the problem.

Problem Topic

The United States Air Force (USAF) provides funding to defense contractors for the development new helmet tracker technology for use in modern war fighters. The USAF is looking to find more methods to independently evaluate the performance of helmet trackers outside of the laboratory environment to further reduce the dependence on contractors to provide performance characteristics. The USAF would like to have more tools available to perform independent performance tests in the field.

Background

The USAF presently utilizes heads-up-displays (HUD) providing pilots with quick access to pertinent information, such as target location and identification. Presently the HUD projects information onto a stationary object in the cockpit so a moving target can drift out of its field of view (FOV), allowing the target to be lost. A solution to the limited FOV of a HUD is a helmet-mounted-display (HMD) that moves with the pilots head, providing the ability to keep the target within the display's FOV even if the pilot needs to look straight up or to the side.

For a HMD to accurately display information, such as target location, a tracker system must be used to determine the location and orientation of the pilot's helmet within the space of the cockpit. The tracker system's performance, speed and accuracy, needs to be evaluated before it can be used effectively by a pilot. The present method to measure static rotational accuracy about three axes requires a cumbersome mechanical device that is not easily transportable. For HMDs to be a capable alternative to HUDs, an accurate, reliable, and portable method to measure static rotational accuracy must be devised.

Criteria and Parameter Restrictions

The system resulting from this project will need to be portable. It should be able to be transported by one person to other labs or a flight line; this will provide the ability for on-site testing of other tracker systems. The system must also be affordable, minimizing the use of any custom hardware or equipment. The system will also need to be capable of providing more accurate static rotational angle measurements than current high end tracking systems so it can be used as a standard to base their measurements on. The system will need to demonstrate the capability to measure angles similar to the rotation of a pilots head in a cockpit. This will provide a method of determining if a tracker's accuracy is consistent through a large range of motion.

Methodology

This project was developed in several steps. The first step involved finding a way to determine the 3D position of points using only a 2D image. Next a way of rotating the 3D points relative to a fixed focal point had to be developed. These processes then were

brought together in an algorithm capable of determining the rotations that occurred on an imaging device between two images.

The algorithm then was tested using a software simulation. A test program was developed that could produce images of a target when viewed at different precise angles. Sets of images were obtained showing various rotations of the camera, the image data was then processed using the algorithm and the error between the known rotation and the calculated rotation was observed. Under the ideal conditions of the software test the algorithm was proven to have acceptable performance.

A camera was then purchased to allow for real-world tests of the algorithm involving hardware. The camera was attached to the existing rotation measurement device and another set of data was collected and ran through the algorithm to verify its validity in real-world applications.

Primary Purpose

This thesis presents the results of this investigation.

Overview

The following chapters will show in detail the steps of developing and verifying the algorithm used for image-based rotation tracking. Chapter II contains all the information on the development and operation of the algorithm. Chapter III describes the software simulator, simulation test setup and simulation test results. Chapter IV will present the hardware used for the real-world test, the test setup and the test results. Chapter V will contain the conclusions made based on the results of the tests, a summary of the test results, and recommendation for future improvements to the system.

II. ALGORITHM

Chapter Overview

This chapter will describe the algorithm used to determine the position of the camera. First, the required known parameters will be described. These include target size, shape and original orientation. Next, since the 3-dimensional space (3D) positions of target points needs to be known, the process of obtaining 3D positions from a 2D image will be explained. The process of comparing the positions of the points and determining camera position will be described. Finally the implementation and execution of the process will be presented.

This chapter will begin making references to the imaging devices coordinate system. Since this information will also be referenced in other chapters it is available in the Appendix.

Required Known Parameters

There are a few parameters that must be met for the algorithm to be able to determine the rotations of the camera. First, a target of known dimensions must be in images that will be processed by the algorithm. Second, a calibration image must be obtained. The calibration image will show the target without any camera rotations. Finally, the field of view of the imaging device must be known.

The shape and dimensions of the target are required for comparison calculations. The distances between points in the rotated image will be different from those in the

calibration image. The algorithm will examine these differences and determine what rotations have occurred. This will be discussed in more detail later in the chapter.

For this system an equilateral triangle with a point in the center was chosen as the target (See Figure 1). In addition to the target shape, one or more of the vertices must be marked so roll rotations larger than 60 degrees can be determined. This target allows the algorithm to solve for rotations using seven constraints. If only three points were used only three constraints could exist, the distance from A to B, B to C, and C to A. Adding point D provides the additional known distances, A to D, B to D, and C to D. Since D is a fourth point, all the points can be checked for existence on the same plane, providing a seventh constraint. Seven constraints allow for much better accuracy than three constraints without adding too much complexity to the algorithm. A square target could also have been used with the same results. However, having a point in the center of the target allowed for easier alignment of the imaging device when obtaining the calibration image.

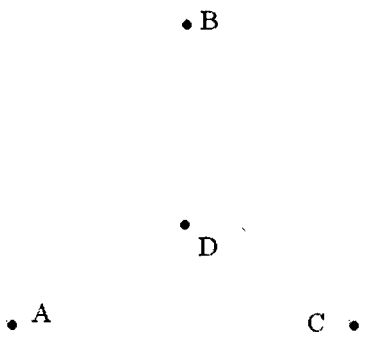

Figure 1. Target.

Though the algorithm is capable of handling slight misalignments, the calibration image is preferably an image obtained with the camera facing the target head on. The algorithm relies on the points in the calibration image to all lie in a plane parallel to the imaging surface. Also, any movement of the imaging device either towards or away from the target (z-direction) will require a new calibration image. While the algorithm can handle the slight distance changes associated with pure rotation, translational movement towards or away from the target is not compensated for. With extra steps, the algorithm can handle movement left, right, up and down (x and y directions), but the primary focus of this system is to determine rotations of the camera.

Finally, the FOV of the camera needs to be known. This information is required for distance calculations involving trigonometry. In software tests, the field of view is setup by the user. When using hardware, the field of view must be determined. A method for this will be presented in Chapter 4.

In summary, this image tracking system requires a target with a known shape and size, a calibration image for every translational position of the camera in 3D space, and a known FOV of the imaging device.

Obtaining 3D Points from a 2D image

Imaging devices are only capable of providing a 2D record of what they see. However, this algorithm requires information on the 3D positions of the target points. Once the 3D positions of the points from both calibrated and rotated images are found the points can be compared and the rotations determined.

Image coordinate system

The images provided by the imaging device have a coordinate system determining the positions of each pixel. The upper left most pixel is designated as position (0, 0). The pixel below this is designated as (0, 1), so the pixels are assigned positions in the format x, y. Later in this chapter the origin of the coordinate system will be reassigned to the center of the image for simplicity. The coordinate system will also be inverted vertically, so that a pixel above the origin will have a positive y-value.

Calibration image

The requirements of the calibration image allow simple calculations to determine the locations of the points in 3D. Since the image has no camera rotations the target plane is parallel to the imaging surface, therefore the target points are all the same distance from the imaging surface. This distance can be determined by performing calculations involving the known dimensions of the target, the FOV of the imaging device, and the distance between the points in the image in pixels.

The image's unit of length is pixels. However to provide real-world measurements the pixels must be converted to an actual unit of length, inches were chosen for this project but any other unit of length could be used. To make this conversion, the length of the line between points A and C in pixels is divided by the known length of the side of a target in inches as shown in Equation 1, which is the Euclidean distance normalized to the target size. The results provide a number with the units pixels/inch that can be used as a conversion coefficient. The inverse of this value (inches/pixel) can be used to determine the x and y positions of points in inches. In Equation 1, Ax and Ay correspond to the x- and y-values of Point A in the same plane.

The same convention is used for Point C and will be used for all points in the rest of this paper.

$$pixels/inch = \frac{\sqrt{(Ax-Cx)^2-(Ay-Cy)^2}}{known\_target\_size}$$

Equation 1. Finding pixels/inch coefficient.

Now that a way to convert pixels to inches as been established, the distance of the camera from the target can be determined. First, the width of the image in inches will need to be determined. The position of center Point D in the calibration image will provide a marker for the center of the image. The left most edge of the image corresponds to a value of zero, so the x position of Point D is half the width of the image in pixels. This pixel value can than be converted to inches using the inches/pixel conversion coefficient.

Next, the angle at the focal point between the center and the edge of the image must be determined. Since the calibration image has the target centered, the angle between the edge and center is simply half the FOV. Using these values and simple trigonometry the distance of the imaging device from the target can be determined. This equation is shown as Equation 2 and illustrated in Figure 2.

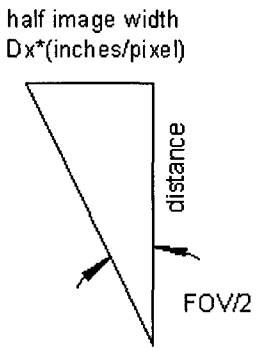

Figure 2. Angles and lengths used to find distance from target.

$$dist = \frac{Dx * (inches / pixel)}{\tan(FOV / 2)}$$

Equation 2. Calculation to find distance from target.

With the distance information along with the pixels/inch coefficient the 2D positions in pixels can be converted to 3D positions in inches. The 2D coordinates are multiplied by the inches/pixel coefficient providing the x, y coordinates in inches. These coordinates are referenced from the upper left corner of the image, as described earlier in this chapter. Now, the coordinate origin is translated to the position of Point D. This is done by subtracting the values of Point D from each point, and multiplying the y-values by negative one to invert the vertical axis. Point A is now represented by (Ax-Dx), -(Ay-Dy), and likewise for other points. The z coordinates for each point are set to the distance determined by Equation 2.

Rotated image

Now that the 3D coordinates of the calibration image are known, the 3D coordinates of the rotated image must now be determined. The process for doing this

with the rotated image is far more complicated than for the calibration image. The first step is to convert all the 2D point positions of the rotated image to inches. This conversion is done the same way as it was for the calibration image, multiply x and y values by the inches/pixel coefficient and set the z values to the distance from Equation 2. This is where the rotated image process becomes more complex.

Unlike the calibrated image where all the points of the target are on a plane parallel to the imaging surface, once a rotation of the imaging device has occurred, the target plane and the imaging surface are no longer parallel. The points extracted from the rotated image therefore do not represent the actual positions of the target points but a projection of those points onto a plane parallel to the imaging surface. Since the points have been projected to an imaginary plane parallel to the imaging surface the distances between the points are no longer the same as the known target parameters.

This effect is demonstrated by concept called "lines of perspective", where points begin to converge to a single point as they fall increasingly further from the imaging device. An example of this effect is looking at a square image such as a computer monitor both head on and from an angle. When viewing the monitor head on it will appear rectangular in shape. View the monitor at angle off center and it will begin to gain a trapezoidal appearance, but the actual shape of the monitor has not changed. The monitor must be viewed at a large angle from head on for this effect to be observed by the naked eye. However, a modern imaging device can resolve minimal deviations from head on.

The distances between points in the rotated image are calculated using the distance between points equation shown as Equation 3. To find the actual positions of

the points in 3D space, the points will need to be moved so that the distances between points are the same as the known target image.

$$dist = \sqrt{(x1 - x2)^2 - (y1 - y2)^2 - (z1 - z2)^2}$$

Equation 3. Calculation of distance between points.

Movement of the points certainly cannot be arbitrary. The points can only be moved along a vector extending from the focal point of the imaging device to the 2D points acquired from the rotated image. Figure 3 shows an example of this. The 2D triangle represents the target derived from the rotated image. The lines from the focal point to the 2D triangle represent the vectors between the two points that the points can be moved along. The 3D triangle represents a target with the same dimensions as the known target dimensions.

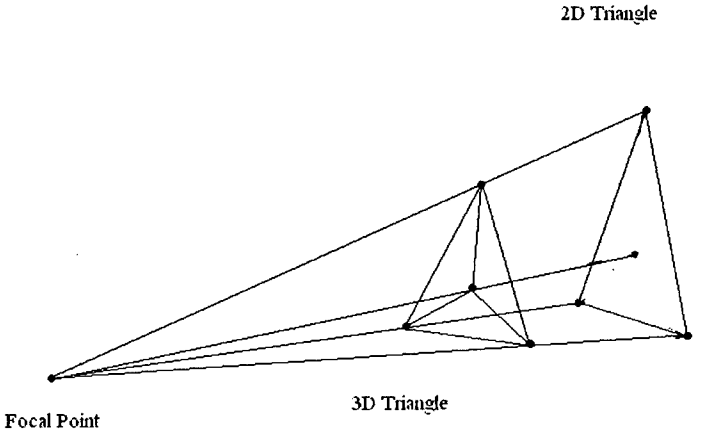

Figure 3. Target projected from 2D to 3D.

To move the points along the vector a coefficient must be used to change each of the three coordinate values (*x, y, z*) the same amount. Each point must have its own coefficient. For example, Point A from the rotated image has coordinates (1,1,1), to move Point A further from the focal point along the vector a coefficient, *j*, is used. If *j* is given a value of 1.5 and used to multiply the three coordinates of Point A, the result is a new projected Point A at (1.5, 1.5, 1.5). If Point A needs to be closer to the focal point j would be set to a value less than one. Each point must be assigned its own coefficient so they can move along there respective vectors independently from the other points. The assignment of coefficients is shown in Table 1.

Table 1

Point Coordinates and Their Corresponding Coefficients

Point	Coefficient	Image Coordinates	Projected Coordinates
A	j	Ax, Ay, Az	j·Ax, j·Ay, j·Az
B	k	Bx, By, Bz	k·Bx, k·By, k·Bz
C	l	Cx, Cy, Cz	l·Cx, l·Cy, l·Cz
D	m	Dx, Dy, Dz	m·Dx, m·Dy, m·Dz

By changing the values of the coefficients the points are moved along the vectors until the distances between each other are the same as in the calibration image. Also the points are checked to make sure they all occur on the same plane. A set of coefficient values is considered valid when the distances between the points match the 6 distances between points in the calibration image and all points are found to exist on the same plane. The test for the distances uses the same distance formula as Equation 3. To test that the points are coplanar, the points are arranged in a matrix as shown in Equation 4

(Weinstein, 1999). If the determinant of this matrix is found to be to zero, the points are coplanar.

The set of coefficients multiplied by the point coordinates derived from the rotated image results in a set of projected points that represent the actual location of the target points in 3D space. These projected points can then be compared with the calibration image points to determine the rotations. For simplicity, the projected point coordinates are now referred to in equations without using the leading coefficient.

$$\begin{vmatrix} Ax & Ay & Az & 1 \\ Bx & By & Bz & 1 \\ Cx & Cy & Cz & 1 \\ Dx & Dy & Dz & 1 \end{vmatrix} = 0$$

Equation 4. Coplanar matrix.

Finding Rotations Occurring Between Two Images

Using the projected coordinates found in the previous step, the rotation that occurred between the calibration image and the rotated image can now be found by using rotation matrices. The points from the calibration and rotated targets must be arranged in a single position matrix with each column containing the (x, y, z) coordinates of the appropriate point. The fourth row of the matrices must be filled in with ones as shown in Equation 5, it a transpose of the matrix used in Equation 4.

$$\begin{bmatrix} Ax & Bx & Cx & Dx \\ Ay & By & Cy & Dy \\ Az & Bz & Cz & Dz \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Equation 5. Position matrix to use with rotational matrices.

Putting the point coordinates in this type of matrix allows them to more easily be operated on by rotation matrices. Rotation matrices allow a way of calculating how points move in 3D space when subjected to rotations about a fixed point, in this case the focal point of the imaging device (Fosner, 1996, pp. 77). The rotation matrices used for the system are shown in Equation 6. The angle θ is pitch, ϕ is yaw, and α is roll.

$$Rx = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \sin\theta & 0 \\ 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad Ry = \begin{bmatrix} \cos\phi & 0 & -\sin\phi & 0 \\ 0 & 1 & 0 & 0 \\ \sin\phi & 0 & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$Rz = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 & 0 \\ -\sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Equation 6. Rotation matrices for X, Y, and Z axes.

The matrices are designed to be used to determine the apparent location of a point in space after a rotation of a known angle has occurred on the imaging device. To do this the coordinates of the point must be in a matrix with the same format as a single column of Equation 5. The rotation matrices can only operate on one point at a time, so only a single column of the matrix is used at a time, this column will be referred to as the point

matrix. The point matrix is then multiplied by the rotation matrices with specified angles in a specific order. The result of the calculation is the apparent position of the point relative to the imaging devices after it has been rotated by the angles specified in the rotation matrices. The order of the matrix multiplication is crucial to obtaining the right results (Fosner, 1996, pp. 79-82). The point matrix is first multiplied by the roll matrix (*Rz*), then the yaw matrix (*Ry*), and finally the pitch matrix (*Rx*) as is shown in Equation 7.

$$Rt = \begin{bmatrix} Ax \\ Ay \\ Az \\ 1 \end{bmatrix} * Rz * Ry * Rx$$

Equation 7. Matrix multiplication order.

The matrix calculation must be performed for each column of the position matrix in order to find the apparent positions of all the points in the target. This is done by changing the contents of the point matrix to the other columns of the position matrix. Equation 7 shows an example using the coordinates of Point A. *Ax*, *Ay*, and *Az* are then replaced with *Bx*, *By*, *Bz* to find the position of Point B. This is repeated for points C and D as well. The results of the calculations of the four points make up the coordinates of the points of the apparent target.

For this application, the points of the apparent target are known and the angles are now the missing variable. To solve for the angles, different angles of rotation are entered into the calculation. If the results of the calculation are the same as the coordinates of the

calibration target points the rotations that occurred on the imaging device have been successfully found. This is a long repetitive process if done manually but produces the necessary results. When solving for the appropriate angles a computer based solving algorithm needs to be employed, this will be discussed in a following section of this chapter.

The rotation required to bring the points to the positions of the calibration target points will be the opposite of the rotations that originally occurred on the imaging device to produce the rotated image. Once the rotations are found they can be multiplied by negative one to provide the rotations that occurred on the imaging device. This is the final output of the algorithm, the yaw, pitch and roll that occurred on the imaging device between two images.

Increasing Range of Tracking

Assuming Point A of the target is differentiated from the other target points, the imaging device can undergo a full 360 degrees of roll rotation, but the range of yaw and pitch rotations the system can measure are dependent on the field of view of the imaging device. For example, the simulator described in Chapter III was setup to have a FOV of 45 degrees horizontal and 35 degrees vertical. The maximum rotation it could track a one inch target from 12 inches away was $\pm$19 degrees yaw and $\pm$ 14 degrees of pitch. The tracking range is smaller than half the FOV to assure that Points A and B are still present. By moving the imaging device further from the target or by using a smaller target the tracking range could be increased a small amount. The absolute maximum tracking range using a single target will be $\pm$ half the horizontal and vertical FOV's of the device for

yaw and pitch respectively. This tracking range is not sufficiently large to be able to track the movements of a pilot's head.

To increase the tracking range of the system multiple targets can be used. The targets are arranged a known distance from each other in a grid pattern (See Figure 4).

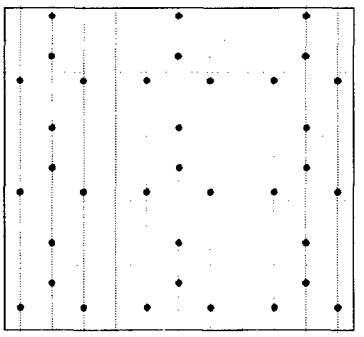

Figure 4. Target grid example.

The positions of the auxiliary targets can be calculated using the known distance between targets and the coordinates of the center target obtained from the calibration image. By using creating a 3 x 3 grid of targets the tracking range can be increased to $\pm$ 35 degrees yaw and $\pm$ 30 degrees pitch using the same setup as the previous example. The auxiliary targets were positioned such that one or more full targets were visible in the image at any given time. The grid can be expanded in size to provide a larger tracking range, theoretically as large as 180 degrees for both yaw and pitch.

Correcting for Camera Misalignments

In software simulation the imaging device can be precisely located at any position so the calibration image will be precisely centered on the target with no rotations. However, when using a real world imaging device such as a camera, it extremely difficult

for such precision alignment. The algorithm needs to be able to accommodate minor variations from the ideal situation.

The algorithm is capable of handling slight rotational misalignments. Since the algorithm compares two images, the calibration image and the rotated image. The rotations that occurred between the two can be calculated, provided that the rotational misalignment was not large enough to cause a significant variation in the distances between each point and the imaging device when the calibration image was acquired. The accuracy of the distance calculation is dependent on the resolution of the imaging device, if the differences of the z-coordinate of the calibration points is less than the smallest distance the imaging device can resolve the algorithm will process the data as expected. There will be a slight deformation of the calibration target, a side of 1" may now be calculated as 1.01", but this discrepancy will be carried through all the calculations and eventually become irrelevant to the final results.

Translational misalignments can be compensated for by adding or subtracting the movement from the coordinates of the points. Assuming the only rotational misalignments are small enough to be neglected, the translation misalignment can be calculated. The resolution of the imaging device must be known, this can easily be determined from the image file. From the resolution the center of the image can be found, it will be at the pixel with the coordinate value of half the horizontal and half of the vertical resolution of the camera. Using the standard calculations, the coefficient of inches/pixel can be found from the off-center target. The offset between Point D of the target and the known center of the image will be the translational misalignment. The

translational misalignment can then be subtracted from both the calibration and rotated points and the rest of the algorithm will provide the expected results.

Algorithm Software Implementation

For the algorithm to be useful, software implementation was needed. There are calculations too complex for the algorithm to be executed by hand. Microsoft Excel was chosen to implement the algorithm because of its capability to handle matrices, the built in equation solver that is required for both the 3D projection and rotation calculations, and the ability to record macros to allow for automated data processing (Microsoft Office Online: Excel, 2005).

The process the Excel spreadsheet performs will now be examined. First, the coordinates of both the calibration and rotated targets, the length of the side of a target, and the FOV of the camera need to be entered into the spread sheet. As soon as these are entered all the required calculations are performed automatically, these calculations are performed in the following order: (1) find the coefficient to convert pixels to inches, (2) find the distance of the camera from the target, (3) find the actual distances between the points in the calibrated target.

After these calculations, the Excel Solver is ready to be run the first time. Solver uses the Generalized Reduced Gradient nonlinear optimization code that can intelligently solve any generic system of equations with numerous constraints (Microsoft Office Assistance: About Solver, 2005). This execution of Solver will adjust the projection coefficients from Table 1 in an attempt to project the points of the rotated image so they are the same distances apart as the calibration target. Solver is not capable of solving the equation with enough resolution to reach the exact distances, but distances within 0.004"

are always returned. This was found to provide accurate enough results which will be described in Chapter III. As an added constraint to further increase the accuracy of the results, Solver also checks that the four projected rotated points occur on the same plane by attempting to minimize the determinant of their matrix (Equation 4).

Once all the constraints are met a matrix containing the projected rotated points is provided. This matrix is then used by the second execution of Solver. This time Solver is set up to adjust the angles of rotations. As the angles are changed the projected points are operated on by the rotation matrices resulting in the apparent position of points at the current angle being tested. These rotated points are then compared with the known points of the calibration target. Solver will stop executing when the error between the rotated points and the calibration points is minimized; the angles that meet this condition are the angles that the imaging device was rotated between images.

Process Execution

This section will walk thru the steps the user needs to take in order to successfully obtain results from the system. The steps are finding target point coordinates, entering them into excel, and executing an Excel macro.

Before the Excel program can be used, the coordinates of the target points need to be determined. To do this a program called ImageJ is used. ImageJ is a freeware image processing application provided by the National Institutes of Health. (ImageJ, 2004) For this system ImageJ is used to find the center pixel of the target points.

After opening an image file in ImageJ a threshold operation is performed. The threshold operation converts the image to black and white, the background will be black and the points will be white. The next operation performed is called center of mass. This

will look at a selected area of the image and determine the coordinates of the "center of mass". White is given the most mass and black is given no mass.

When an area of the image is selected that contains just one point, the center of mass operation will return the coordinates of the center of the point. Each of the four target points in an image should be selected individually and the center of mass found. ImageJ is then capable of storing this information in text file that is used with Excel in the next step. This entire process is repeated for both calibration and rotated images.

The text file provided by ImageJ is opened in Excel. The data is then copied and pasted into the appropriate cells of the Excel program. The final step is executing the macro included in the Excel program. This macro will run Solver, move the resulting data to new cells, and execute the second Solver which will output the rotations that occurred between the two images.

III. SIMULATOR

Chapter Overview

This chapter will describe in detail the software simulator, the simulation test setup, and the results of the simulation test. The first section will describe the simulator itself including what programming language was used to write it and supply information on what preexisting packages were used during the development of the code. The next section will describe the functionality of the simulator as well as show examples of the simulator output at different rotations. The following section will describe the setup used for the simulated test. The final section of this chapter will present the results from the simulated test and compare them with the tracker accuracy of a high-end optical tracker.

Programming and Graphics Languages

The software simulator was written in C++ using the freeware Bloodshed Dev C++ compiler (Bloodshed Software – Dev C++, 2003). C++ was used for this program because of the author's previous experience with C++ code, it is the language most often used at the author's work place and the availability of a powerful graphics package that can easily be used with it.

The graphics package used in the software simulator is freely available project called OpenGL (OpenGL Overview, 2005). OpenGL provides a complete set of functions that allow objects to be drawn in 3D space using C++. The primary attraction

of OpenGL is that it masks the programmer from the high level math required to correctly display 3D objects in space on a 2D computer monitor. Instead of the rewriting code to perform 3D calculations and place pixels in the necessary positions on the screen, OpenGL allows the user to specify the location of 3D objects using a set of 3D coordinates. OpenGL can also set the objects drawn to be either world referenced or camera referenced.

Drawing targets using OpenGL

OpenGL was used in the software simulator to serve as the imaging device capable of generating images that could be analyzed by the tracking algorithm. The first task of the software simulator was to draw the targets. The simulator allows the length of a side to be specified by the user. The target size is then calculated based on entered value. Once the size is known, a target with Point D centered on the origin is drawn. To draw the additional points their x, y coordinates need to be calculated, the first step is determining the distance from the center, Point D, to the other Points A, B and C. Figure 5 and Equation 8 show how this is accomplished using the law of sines.

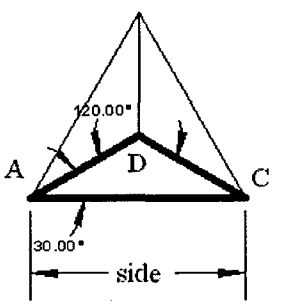

Figure 5. Triangle used with law of sines.

$$dist = \frac{\sin 30 * side}{\sin 120}$$

Equation 8. Law of sines used to find distance.

Using the distance to the center calculated by Equation 8, the coordinates of Point B can be found. Since Point B lies along the same vertical line as Point D, Bx can be set to Dx and By can be set as the distance from the center. Points A and C required additional trigonometry. Figure 6 shows the triangle used to calculate Ay by solving Equation 9. Since Points A and C exist on the same horizontal line Cy is the same as Ay. The Ax and Cx both have a magnitude that is equal to half the length of the side. Since Points A and C lie on different sides of the Origin at Point D, Ax will be negative and Cx will be positive.

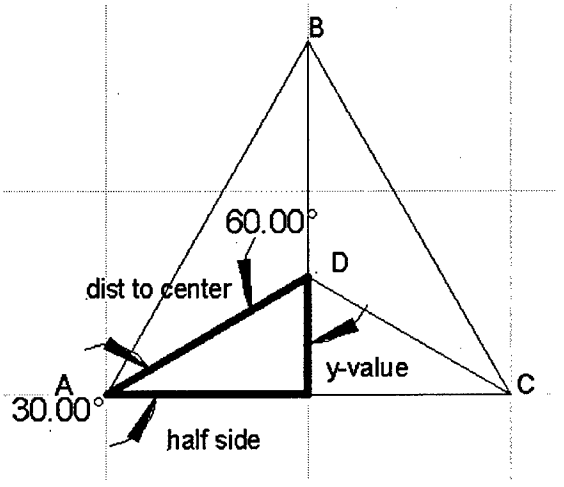

Figure 6. Triangle used to find the Ay.

$$Ay = \sqrt{dist^2 - (side/2)^2}$$

Equation 9. Pythagorean theorem used to find Ay.

Table 2 provides the values of each point of the target in terms known or already calculated values: distance from center, Ay, and side length. The z-values are the same for each point since they all exist on the same plane.

Table 2

Point Coordinates Calculated From Length of a Side

Point	X	Y
A	-side/2	$Ay = \sqrt{dist^2 - (side/2)^2}$
B	0	$dist = \frac{\sin 30 * side}{\sin 120}$
C	side/2	$Ay = \sqrt{dist^2 - (side/2)^2}$
D	0	0

Now that the primary target points all have coordinates, a grid of targets can be setup to allow for a greater tracking range (See Chapter II). The spacing between targets was determined by experimentation. A value was chosen small enough that there were no gaps in the tracking range where an entire target was not visible, but large enough that the targets were far enough apart to provide a substantial increase in tracking range. It was then observed as the camera was moved different distances from the target plane that different grid spacing would be required based on the distance from the target. As the camera was moved close to the target the spacing between targets was too large. Too far away and the spacing was too small. A grid spacing coefficient was then determined by dividing the known good grid spacing divided by the distance of the camera from the target. This value was found to be 0.28. Whenever the simulator is executed the grid spacing is determined by multiplying the default camera distance value by the grid

spacing coefficient. This provides a grid of targets with optimal spacing for a large continuous tracking range.

Using the grid spacing coefficient with a pattern of five columns and three rows of targets, a tracking range of $\pm$ 50 degree yaw and $\pm$ 30 degrees pitch was obtained. This range can be easily increased by adding more rows and columns. For this simulation the grid spacing was added to test that the algorithm would work as expected when using multiple targets. As the angle of tracking becomes greater the system will lose some accuracy though. As the target points are viewed at a greater angle the points will appear closer together decreasing the measuring resolution of the imaging device. However, in the ranges tested there was little degradation of tracking performance as will be presented in the Test Results section.

Postioning the camera in OpenGL

Once the targets are drawn the simulator moves the camera position to the specified starting distance with the camera FOV centered on Point D of the center target of the grid. The simulator now allows the camera to be translated and rotated to any position required. OpenGL provides simple function calls to make these movements. For example to rotate the camera to 5 degrees on the x-axis the command "glRotatef(5.0f, 1.0f, 0.0f, 0.0f);" is used, where the 5 represents the angle to rotate to and the 1, 0, 0 represents the vector to rotate about (Fosner, 1996, pp. 96). The translation functions operate in a similar fashion.

It is important to note that as the camera is moved the targets appear to move while in fact they are stationary, world referenced objects. The position and rotation of the camera is also world referenced, however the image displayed on screen is

determined by the camera's field of view and orientation. This is what allows OpenGL to simulate images provided by a real world imaging device. A real world camera also moves relative to object in the real-world and can only view what falls in to the FOV at a given position and orientation.

Now that the targets have been setup in 3D space and the camera can be positioned precisely, the simulator can be used to perform the first validation tests of the algorithm.

Simulation Test Setup

This section will describe how tests using the software simulator were planned and executed. There were two sets of tests performed with the simulator. The first set used only a single target. The objective of this test was to test the basic performance and validity of the algorithm. The second test involved using a target grid with an increased tracking range. The goal of this test was to determine if using a grid of targets degraded the performance of the algorithm at all.

Single target test

The simulator displayed only a single target centered on the origin. The test was setup to test rotations on each axis individually plus a combination of two and three axes. Eight cases were established as shown in Table 3. The cases were chosen so there would be a positive and negative rotation for each axis. This series of test cases was run several times with different distances between the camera and the target. These results will be presented in the Test Results section.

Table 3

Rotation Values for Single Target Test Cases

Case	Yaw	Pitch	Roll
0	0	0	0
1	0	0	-17
2	8	0	0
3	0	-4	0
4	-3	0	8
5	6	6	0
6	0	-3	-3
7	5	5	5

Target grid test

For these tests the simulator displayed a grid of targets with 5 columns and 3 rows. This test was again setup to test each axis individually and combinations of two and three axes. The rotations angles of this test were greater than those of the single target tests and multiple targets had to be used to obtain solutions. Table 4 shows the label given each target and its position relative to the center target 0.

Table 4

Target Grid Labels

	Column 1	Column 2	Column 3	Column 4	Column 5
Row 1	1	2	3	4	5
Row 2	6	7	0	8	9
Row 3	10	11	12	13	14

Table 5 contains data representing the rotations and target used for each case in the Target Grid test. Each case was only run once with a distance of 12" and a target size of 1".

Table 5

Rotation Values for Target Grid Test Cases

Case	Target	Yaw	Pitch	Roll
0	0	0	0	0
1	6	23	0	0
2	12	0	-20	0
3	0	0	0	30
4	0	-3	0	30
5	4	-15	17	0
6	12	0	-22	-5
7	2	15	15	15

Recording data

The simulator was set to the position and rotations for the appropriate test case. The image displayed by the simulator was then saved as an uncompressed bit map to avoid any errors that could be contributed by a compression algorithm. The saved images were then processed in ImageJ and the resulting values entered into the Excel program outputting the angles determined by the algorithm.

The output images for both trials were set to have a FOV of 45 degrees horizontal and 35 degrees vertical resulting in a an image resolution with a resolution of 1260 x 980. This size was a limitation of the monitor that was used to run the simulations. A higher resolution image would provide more accurate results. Based on the pixel size in an image of 1260 x 980 the highest precision that can be expected is 0.036 degrees per pixel. This was calculated by dividing the horizontal FOV (45 degrees) of the imaging device by the number of horizontal pixels in the resulting image (1260).

Test Results and Comparions

This section will present the test results of both the single and grid target tests. The accuracy information on a high-end optical tracker will be provided so that comparisons between the ideal image tracker performance and actual optical tracker can be made.

Optical tracker data

The optical track used for this comparison is called the HiBall developed by 3rdTech Inc. The Hiball uses a 12'x 12'grid of infrared light-emitting diodes (LED's) attached to the ceiling and a set of sensors that can be moved freely in the area underneath the grid. The HiBall tracks by illuminating LED's at known locations and processing which LED's are visible to what sensors to calculate the sensor position and orientation..

The HiBall output varies greatly when stationary so it is difficult to make accurate static measurements. It is also susceptible to noise contributions from room lighting. The sensor was set on a solid stationary surface with the lights on and the HiBall collected 24000 data points. The standard deviation of the 24000 points was then calculated for rotation about each axis: 0.031 degrees yaw, 0.067 degrees pitch, and 0.034 degrees roll. The accuracy of the HiBall for static measurements cannot be better than these values, at any given rotation the points could vary as much as these values.

Single target test results

For this test the image tracking system is using ideal simulated images and is not susceptible to noise such as room lighting. Also, every test is repeatable with no deviation between trials. The software simulator will always produce the identical image

from trial to trial, so there will be no deviation of values at the same orientation. These results will indicate that the error of the new software-based image tracker will be less than the ranges of data output from the HiBall

The results of the single target tests are shown in Table 6. An observable trend is that the further the target is from the camera, the less accurate the measurements are. However, being closer to the target has the undesirable effect of reducing the tracking range. Based on the results the best trade off between accuracy and tracking range occurred at a distance of 12". The tracking range at 12" was $\pm$ 19 degrees yaw and $\pm$ 14 degrees pitch determined by experimentation. The best performance was at a distance of of 6", but at this distance there was only a range of tracking of $\pm$ 17 degrees yaw and $\pm$ 12 degrees pitch, also determined through experimentation.

Table 6

Average Error at Varying Distances

Distance(inches)	Average Yaw error (degrees)	Average Pitch Error (degrees)	Average Roll Error (degrees)
3	0.011	0.038	0.042
6	0.010	0.012	0.10
9	0.012	0.015	0.16
12	0.015	0.016	0.15
36	0.031	0.040	0.65

The data from Table 6 shows that in optimal conditions that the image tracker is capable of providing results with less error is the yaw and pitch directions than can be expected by using the HiBall. It is also important to note that the yaw and pitch values are smaller than the expected precision of 0.036 degrees determined by the image

resolution. However, the large error on the roll is the cause of some concern. The algorithm handles all three rotations the same computationally, so a possible contributor to this error could be how OpenGL handles roll. To determine if the algorithm or simulator is responsible for the roll error the results of the hardware test will be examined in the next chapter.

However, based on the yaw and pitch accuracies the image tracking algorithm can function as a static rotation tracker with higher accuracy than existing high-end tracking equipment.

Target grid test results

Now that the tracker algorithm has been validated when using a single target the results of using a target grid with increased tracking area will be examined. The following data in Table 7 will show that by using the target grid the image tracker is still capable of providing more accurate results than the HiBall. It can also be noted that the average error is still not much greater than the expected precision of 0.036 degrees based on image resolution of the simulator and pixel size.

Table 7

Average Error at 12" with Target Grid

Targets	Average Yaw error (degrees)	Average Pitch Error (degrees)	Average Roll Error (degrees)
Single	0.015	0.016	0.15
Multiple	0.015	0.039	0.37

Simulation Test Conclusions

The results of the simulation test show that the image tracking algorithm, in ideal test situations, is capable of producing angle measurements in yaw and pitch rotations less susceptible to error than the HiBall tracker. The test results also validate the tracking algorithm, since in most instances the algorithm errors were smaller than minimum amount of error due to pixel size.

IV. HARDWARE

Chapter Overview

This chapter will describe how the algorithm was tested using hardware as opposed to software. The hardware test allows for greater error due to camera alignments no longer being perfect and the noise in the images caused by compression algorithms.

First, the specifications of the test hardware will be presented. The two most important pieces of hardware for the test are the gimbal and the camera. After that, measurements on the hardware will be described such as finding the center of rotation of the gimbal and the FOV of the camera. Then, measurements on the camera and the mounting procedure will be described. Next, the test setup will be reported followed by a section presenting additional processing to account for translation offsets and finally the test results.

Gimbal Specifications

The Gimbal used for this test is capable of rotating three axes about a single point in space (See Figure 7). The gimbal is capable of a full 360 degrees yaw rotation, $\pm$ 25 degrees pitch, and $\pm$ 80 degrees roll with one-minute graduations on each axis. The designer of the Gimbal was contacted and reported that three axes of rotation intersect at a point within 0.05" of each other. This means that the gimbal itself will not contribute significant error to the results. This gimbal has served as a test instrument to perform

rotational measurements in the lab. The down side of the Gimbal is that it is large and cumbersome; at least two people are needed to move it.

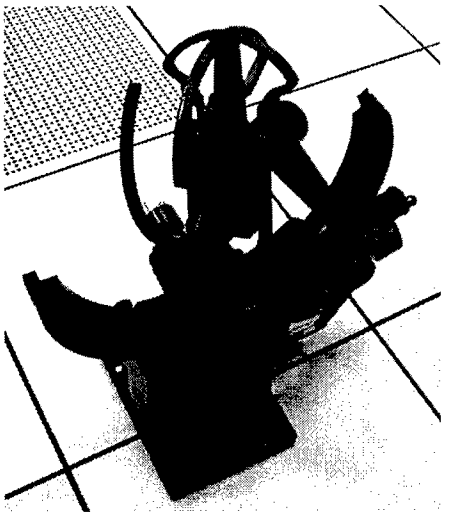

Figure 7. Gimbal used in hardware test.

Camera Specifications

The Camera selected to use for the hardware test was a Canon Digital Rebel 300D. This camera has a high resolution of 3072 x 2048, providing a degrees/pixel value of 0.015 degrees allowing the algorithm results to have higher resolution. Another important feature was the ability to remotely release the shutter to eliminate movement cause by pressing the shutter button. The Canon allows remote shutter release through USB by using the provided software. The Canon also has the ability to turn off auto focus and focus the camera manually. As the camera is rotated to different angles auto-focus may focus on an object further away, blurring the part of the image containing the target. A blurry target would reduce accuracy by adding more noise into the system, so having the camera always focused on the target reduces noise contribution. The rotations

performed during the hardware test were not large enough to be effected by a limited depth of field of the camera. In all the acquired images the all target points were sharp. Finally, the tripod mounting hole on the bottom of the Canon is also closely aligned to the focal point of the lens, reducing misalignment errors (Canon Inc., 2003, pp. 132-136).

Finding the Center of the Gimbal

Before the camera could be mounted to the gimbal the center of rotation of the gimbal had to be determined. To do this a sphere was mounted to the gimbal. A grid pattern with 1/32"squares with was taped onto the sphere. Figure 8 shows the grid on the sphere. Distortions of the grid are obvious in the photo. Since the grid was not used for any measurements, only as a means of finding the position of a point relative to other features on the sphere the distortions were not a cause of concern. The Canon camera was mounted on a tripod with the sphere centered in the FOV. The gimbal was then rotated about pitch and roll axes at 5 degree increments, at each position an image was recorded. The recorded images were examined and a point that remained stationary in each of the images was found. Since the point remained stationary through the rotation it was determined to be the center of rotation of the gimbal. The distance of this point from the top of the base was measured using a set of micrometers and was found to be 1.22".

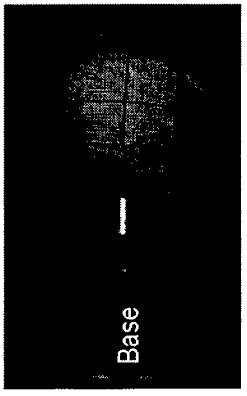

Figure 8. Sphere with grid.

Finding Camera FOV

The FOV of the Camera needs to be determined so distance calculations can be performed from the test images. Calculation of the FOV can be done by rearranging Equation 2 in Chapter 2 to solve for FOV in terms of inches/pixel, Dx and the distance from the target to the focal point (See Equation 10). The distance of the camera from the target must be measured when the picture is taken. Dx and inches/pixel can be determined that same as they were in Chapter 2.

$$FOV = 2 * \arctan\left(\frac{Dx * (inches / pixel)}{dist}\right)$$

Equation 10. Find FOV.

Positioning and Mounting Camera

With the camera mounted on the gimbal, the offsets between the center of rotation of the gimbal and the focal point of the camera need to be determined. First the center of the camera coordinate system must be defined. The lens points along the positive z-axis. When looking down at the camera from above, the camera is being view from along the y-axis. Still viewing the camera from above, if the lens is facing forward the positive x-axis goes to the right. Next, a measurement was made from the bottom surface of the camera to the center of lens. This distance was found to be 1.54" along the y-axis. The center of the gimbal was found to be 1.22", thus there is offset along the y-axis of 0.32". Another measurement was made from the center of the tripod mount to the center of the lens. From this measurement the offset along the x-axis was found to be 0.08"

Finding the offset along the z-axis was less straight forward. The focal point of the camera occurs at some point along the lens pointing along the z-axis. If the camera is rotated about its focal point objects lined up in the FOV will remain lined up as the camera is rotated. Two rods were setup in a line and viewed head on by the camera so one rod was blocking the other rod. The camera was rotated at different points along the lens, when a position was found where the rods remain lined up even after the rotation it was marked as the focal point (Panosaurus Setup Information, 2004). The distance from this point to the center of the tripod mount was measure. This measurement indicated that there was an offset along the z-axis of 3.7".

With all the offsets found the camera was then mounted to the gimbal. A metal rod was used that screwed into the tripod mount on the bottom of the camera. This rod fit in the mounting hole on the gimbal and was secured in place by set screws. The camera's base was positioned flush against the base of the gimbal. A level was used to test that both the gimbal and the camera were parallel to the ground along the x- and z-axes.

Hardware Test Setup

To perform the hardware test, the gimbal and camera assembly was positioned facing a wall. The camera was connected to a laptop with a USB cable to allow for remote shutter release. A 2" target was drawn in a CAD program called Autosketch and printed out. The target was measured with micrometers to confirm that it did in fact have the correct dimensions. The target was taped to the wall with much care being made the target was perpendicular the ground. To test that the target was level a picture was taken with the camera and Points A and C were checked to see if they were positioned on the

same horizontal line. The target was adjusted until the Points A and C occurred on the same line.

The camera was set to take images with a resolution of 3072 x 2048. The image was stored as a JPEG image file, the lower compression option was selected to reduce error contributed by the compression algorithm. The camera had an option to store the images as uncompressed RAW image files. This option was not used because ImageJ does not have native support to read RAW images.

The first picture taken was the calibration image; this was followed by a set of images at various angles of rotation. The points recorded for the hardware test are presented in Table 8. The rotations were chosen to ensure that positive and negative rotations about each axis were tested for. Only a single target was used for the hardware test, the validity of the target grid test was proven during the simulation test. After all images were obtained they were processed by the algorithm, the results are presented in the next section.

Table 8

Hardware Test Rotations

Case	Yaw (degrees)	Pitch (degrees)	Roll(degrees)
0	0	0	0
1	5	0	0
2	0	-10	0
3	0	0	5
4	-5	-5	0
5	-10	10	0
6	-5	5	0
7	-5	-5	-5

Hardware Test Results and Conclusions

The images were processed in ImageJ and the center of the target points found. Since the points in the hardware test images were larger than one pixel, this process could produce some possible error based on the JPEG compression algorithm and the algorithms used by ImageJ. The points from the resulting file were loaded into Excel and the algorithm executed on them. The results were recorded and placed in Table 9.

Table 9

Hardware Test Results

Case	Yaw (degrees)	Pitch (degrees)	Roll(degrees)
0	0.000	0.000	0.000
1	4.960	0.048	-0.020
2	-0.187	-9.955	0.125
3	0.002	0.006	5.022
4	-5.108	-4.960	0.608
5	-10.305	9.769	0.481
6	-4.814	5.038	-0.367
7	-5.562	-4.932	-4.460

The average errors about each axis were 0.174 degrees yaw, 0.060 degrees pitch and 0.270 degrees roll. The percent error for all values was calculated to be 2.4%

The results are not nearly as accurate as the software simulation, but they were not expected to be in the less ideal conditions of the hardware test. Factors such as camera position contributed error to the process. Still, the results indicate that the algorithm is still capable of making measurements using real world hardware. When compared to other rotational errors, roll is not consistently larger as was observed in the simulation results; this indicates that the algorithm does not have problems calculating

roll. It should also be noted that test involving real world hardware require a great degree of care when positioning the camera. From looking at the data it can be seen that there was an induced rotation caused by the focal point not being precisely located at the center of rotation of the gimbal. This can be observed in Case 2 where only a pitch rotation was dialed in on the gimbal but a significant yaw and roll were observed.

V. CONCLUSIONS AND RECOMMENDATIONS

This chapter will discuss how this project has developed a tracking system capable of making static rotational measurements while at the same time being completely portable.

Summary of Results

The results of this project have provided an image based tracker system that is capable of providing a method to test most modern tracker systems. The image tracking algorithm developed by this project was tested against a high-end optical tracker and was able to provide more accurate results of static rotational measurements in ideal noiseless conditions. The tracking algorithm was then tested in less than ideal real-world tests where issues such as misalignment, image compression and offsets from center were encountered. The real world result indicates that the algorithm does work with real world hardware. For the system to serve as a high accuracy measurement device further work would need to be done to compensate for induced rotations caused by the camera not being precisely rotated at the focal point.

Conclusion Based on Results

Based on the results of this project it can be concluded that an image based tracking system can potentially be used to measure the static rotational accuracy of other tracker systems. The entire system requires only a digital camera, a target, and a tripod.

This equipment can easily be moved by one person meeting the portability requirement. The system cost is also affordable and even scalable based on the price of the digital camera used. In addition to the camera the only other equipment needed is a tripod. Any digital camera can be used, this is what makes the cost scalable. Any entry level 2 mega pixel camera can be used if lower accuracy is sufficient for the test. If a higher accuracy is desired a more expensive, higher resolution camera can be used.

Possible Future Improvements

In the future, this project could be expanded to accommodate dynamic rotational measurements. This could be done by using a video camera to record a target during rotations. The output of the video camera could be analyzed providing a motion tracker with a tracking frequency equal to the frames per second of the video recording.

Another solution to eliminate some of induced rotations would be the use a device designed for panoramic photography. These devices allow the focal point of the camera to be positioned at its center of rotation. These devices however do not allow for roll to occur.

As technology progresses the cost of higher resolution digital imaging will decrease allowing the accuracy of this tracking algorithm to increase. As was observed in the simulation tests, the algorithm is capable of tracking movements down to the best possible resolution as determined by the number of pixels in the image.

Future work could also go into integrating computer vision into the process and eliminating the need of the user to pick out the target points by hand. Once this capability is achieved code could be written that would automate the entire process.

REFERENCES

Bloodshed Software – Dev C++. (2003). Bloodshed Software. [Online]. Available: http://www.bloodshed.net/devcpp.html

Canon Inc., (2003) Canon EOS 300D Instruction Manual, (pp. 132-136).

Fosner, R. (1996, October). OpenGL Programming for Windows 95 and Windows NT (pp. 71 – 96)

ImageJ. (2004). ImageJ: Image Processing and Analysis in Java. [Online]. Available: http://rsb.info.nih.gov/ij/

Microsoft Office Assistance: About Solver. (2005). About Solver. [Online]. Available: http://office.microsoft.com/en-us/assistance/HP051983681033.aspx

Microsoft Office Online: Excel 2003 Home Page. (2005) Excel [Online]. Available: http://office.microsoft.com/en-us/FX010858001033.aspx

OpenGL Overview. (2005) OpenGL - The Industry's Foundation for High performance Graphics. [Online]. Available: http://www.opengl.org/about/overview.html

Panosaurus Setup Information. (2004) Step 4:Preparing to Find the Optical Center. [Online]. Available: http://gregwired.com/pano/S4.htm

Weinstein, E. W. (1999). Coplanar [Online]. Mathworld—A Wolfram Web Resource. Available: http://mathworld.wolfram.com/coplanar.html

GLOSSARY

Calibration Image:	An image obtained at any translational position of the imaging device viewing the target head on with no rotations.
Field of View:	The viewable range of an imaging device that spreads from the focal point a set angle. Often the vertical and horizontal fields of view have different values.
Functions:	In C++ functions are section of commonly used code that can be easily called multiple times. Functions prevent the same code from being written several times.
Imaging Device:	Any device capable of producing images used for the tracking algorithm. For this project the imaging device was either the software simulator or a digital camera.
Imaging Surface:	The surface on which an image is made. Film acts as the imaging surface on traditional cameras, while most modern digital camera use a Charge Coupled Device (CCD) sensor.
Point Coordinates:	The three values of a point describing its location in 3D space.
Rotated Image:	An image obtained at the same translational postion as the calibrated image, but with different angles of rotation.
Rotation:	Movement about an axis that changes the orientation of an object.
Tracker range:	The range of motion a tracker can report. For the tracker algorithm this is determined by the FOV of the imaging device.
Translation:	Movement along an axis that changes the postion of the object.

APPENDICES

APPENDIX A

COORDINATE SYSTEM

In this paper movements will be referred to in the direction they occur such as positive x, or negative y. The coordinate system is setup with the origin being the focal point of the imaging device. The positive z-axis extends in the direction the imaging device is facing, the positive x-axis is to the right of the imaging device, and positive y is above as shown in Figure A.

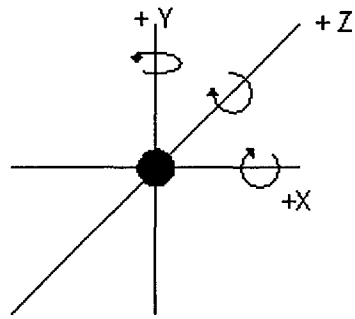

Figure A. Imaging device coordinate system.

Rotations occurring along a specific axis are referred to in this paper as yaw, pitch, and roll. A positive yaw rotation corresponds to a counter-clockwise (CCW) rotation when looking from the positive y-axis toward the origin. A positive pitch rotation occurs when the imaging device is rotated CCW when looking toward the origin along the x-axis. A positive roll is when a clockwise rotation occurs looking from the origin in the positive z direction. The positive directions of rotations are marked by the arrows in Figure A.

APPENDIX B

RELATIONSHIP TO ELECTRICAL ENGINEERING

PROGRAM OUTCOMES

Electrical Engineering Program Outcomes

Program Outcome a. An ability to solve electrical engineering problems by applying knowledge of such fundamental and advanced mathematics as calculus, differential equations, linear algebra, probability and statistics, and science and engineering principles.

> This project required me to apply mathematics, primarily matrix algebra, to create the desired system. Additional tools making use of other advance mathematics were also employed, such Excel's Solver.

Program Outcome b. An ability to design and conduct experiments in electrical engineering as well as to collect, analyze and interpret data to reach appropriate conclusions.

> Part of this project was to test the tracking system that had been developed. In order to test the system, a series of experiments had to be developed and conducted. The data from these experiments was then collected and analyzed. Conclusions were made based on the data collected during the experiments.

Program Outcome c. An ability to design an electrical system, component, or process to meet desired technical, environmental, safety and economical specifications.

> This project produced a system capable of performing rotational measurements of an object using digital imagery.

Program Outcome i. An appreciation for the need, and preparedness to enage in life-long learning.

> During this project many new things had to be researched. Some examples are: 1) how do rotation matrices work, 2) how can numerous equations be solved for at once, 3) how does OpenGL work? To continue improving, you need to continue learning.

Program Outcome k. An ability and experience in using the techniques, skills, and modern engineering tools necessary for engineering practice.

> For this project a technique was used to test the system through simulation before performing a real-world test. This technique is often done by engineers to reduce costs. Simulations are often less expensive to perform than real-world tests. Using a simulation also provides a way to prevent outside factors from contributing error to test results.

Program Outcome l. A knowledge of computer science and computer engineering, and engineering sciences necessary to analyze and design systems containing hardware and software components.

This project relied heavily on computer science in both the algorithm used for tracking and the simulator used to test the program. Code was needed in the algorithm that could perform tedious calculations not possible by hand. The simulator program was written from the ground up for this project.